COMP-4478-WA - Game Programming Project I

I. Game Explanation

The Game is called “The Joker”. It is a Memory Game. At the start of the game the player is shown 12 cards (randomly selected from the pack of 52 cards) placed in a grid and the user is given 10 seconds to remember each card location. After 10 seconds all the 12 cards are turned down. The player then starts opening the question cards given to him/her and for each question card the player has to find a match in the grid. A single wrong match and the player loses the game.

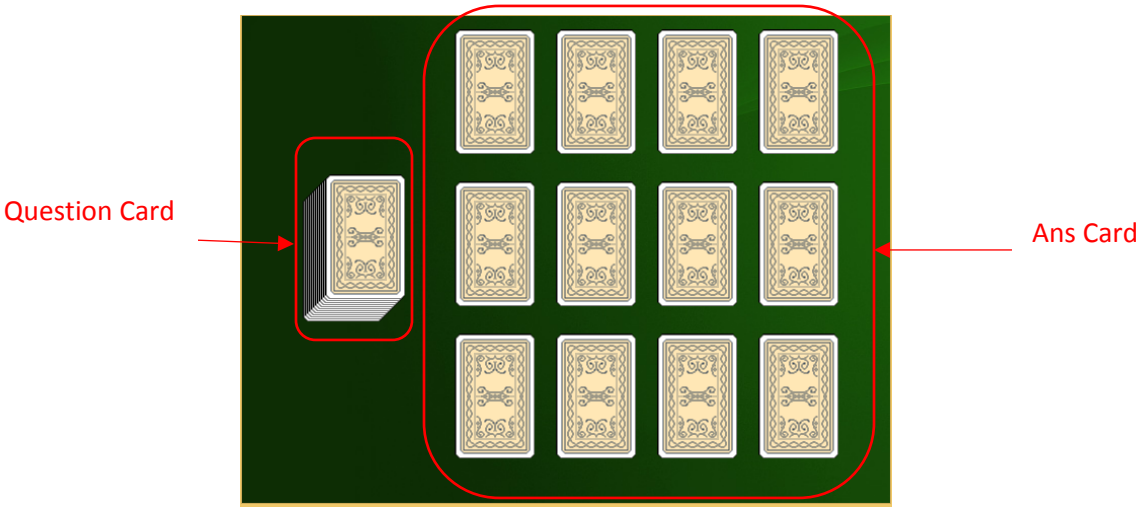


Figure 1: Game Screen

II. How to Play

- Press the joker card on the menu screen
- Wait for 10 second to memorize the card
- Once the card turns down click the question card
- Click the matching ans card

III. Rules

- Match all the ans card with the question card.
- A single wrong match and the player loses the game.
- The player wins on all successful match

Note: To test the wining condition take a photo of the grid and then play

IV. How to Run

1. Add the folder to sublime text project
2. Press Ctrl+Shift+B

3. Select Neko Test
4. Press Ctrl+Enter

V. Program Explanation

Step1: Defining all the Cards on the screen

The project consists of Deck.png in the assets/images directory. This png file consists of all the 52 cards as frames of equal size. Here each of the frame can be accessed with a frame index, with last frame (card top view) having frame index 54.

Figure 2: Deck.png

The 12 grid cards (ans cards) are generated using the below code snippet. The steps are as follows.

```
var pick = FlxG.random.int(0, 51, pickedCards);

pickedCards.push(pick);

cards.add(new Card(x, y, pick,"ansCard"));
```

Figure 3: Card Grid

- A random frame index is chosen between 0 and 51 that is not present in the array pickedCards.
- The chosen frame index is pushed to array pickedCards (to avoid dublicates).
- A object of class Card is formed which includes details regarding the cards location, the card frame index and the type of card (i.e where it is a part of ans card or question card).
- Add the card object to the cards group which is a FlxTypedGroup that stores all the 24 card object (12 ans and 12 question) generated in the game.

The question cards are generate using the below code snippet. The steps are as follows.

```
var questioncardIndex=FlxG.random.int(0,k-1,pickedquestionedCards);
pickedquestionedCards.push(questioncardIndex);

cards.add(new Card(x,y,pickedCards[questioncardIndex],"questionCard"));
```

Figure 4: Question Card

- Here k is the length of pickedCards array.
- A random frame index is chosen from the pickedCards array (which consist of chosen 12 grid cards frame index) that is not present in the array pickedquestionedCards.
- The chosen frame index is pushed to array pickedquestionedCards (to avoid dublicates).
- A object of class Card is formed which includes details regarding the cards location, the card frame index and the type of card
- Add the card object to the cards FlxTypedGroup.

The x and y are cards are constantly modified to position them correctly.

Step2: Showing the cards for 10 seconds and flipping them over

A forEach loop is ran on the crads FlxTypedGroup and for every card object in the group the display function is called. After 10 seconds again a forEach loop is ran on the crads FlxTypedGroup to turn the cards down.

```
public function display(i:Int){
    if(i==0){
        animation.frameIndex = cardIndex;
        turned= false;
        FlxG.sound.play("assets/sounds/cardOpen.wav");

    }
    else{
        animation.frameIndex = 54;
        turned= false;
        FlxMouseEventManager.add(this, onDown, null, onOver, onOut);
        FlxG.sound.play("assets/sounds/cardOpen.wav");

    }
    FlxTween.tween(scale, { x: 1 }, TURNING_TIME / 2);
```

Figure 5:Display Function

The display function takes input as 0 or 1 indicating whether the card has to be turn up or down and then the animation of turn is performed using FlxTween. Once the card has be turned down again the MouseEventManager is enabled for each of the card thereby making the cards clickable.

Step3: Handling Mouse click on cards

Each card on the player screen is clickable so the player can turn the card on a click of a mouse. A MouseEventManager is placed on each Card as shown below. Here the main function is onDown

```
FlxMouseEventManager.add(this, onDown, null, onOver, onOut);
```

Figure 6: Mouse Event

- When the mouse is clicked on the card the onDown function is called
- When the mouse is hover over the card the onOver function is called.

The onDown Function is defined as below. Here on each click of mouse on the card FlxTween animation is performed for turning the card and then pickCard function is called.

```
function onDown(_)
{

    if (!turned)
    {
        turned = true;
        FlxTween.tween(scale, { x: 0 }, TURNING_TIME / 2, { onComplete: pickCard });
    }

}
```

Figure 7:On Click of mouse

The function pickCard

- First it is checked if the card is a questioncard. If so the questioncard is saved and the flag isquesionCardOpen is set to 1 indicating the question card is now open .

```
if(cardType=="questionCard"){

    FlxTween.tween(scale, { x: 1 }, TURNING_TIME / 2);
    PlayState.isquestionCardOpen=1;
    PlayState.questionCard=this;
    FlxG.sound.play("assets/sounds/cardOpen.wav");
}
```

Figure 8:Event Handling for question card click

- Second it is checked if it is ans card and the questioncard is open. If the question card is open it checks whether the question and ans card matches if they do not match the game ends. However, if they match then both the cards are killed and a counter variable is incremented. When the counter variable reaches the count of 12 it indicates the game is finished.

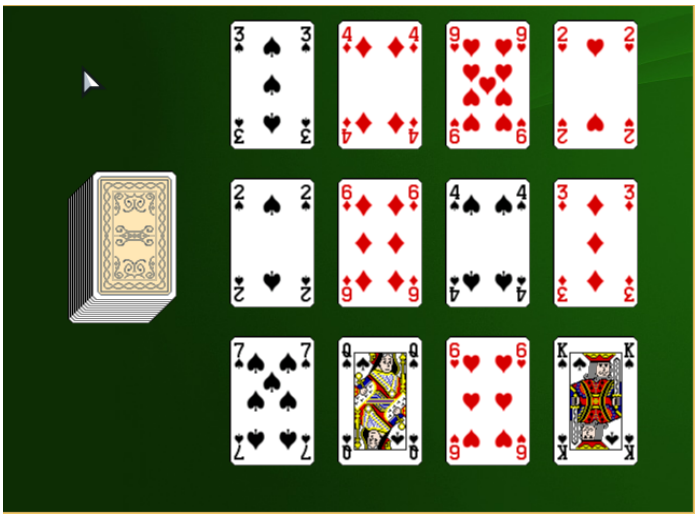
```
else if(cardType=="ansCard" && PlayState.isquestionCardOpen==1){
    FlxG.sound.play("assets/sounds/cardOpen.wav");
    FlxTween.tween(scale, { x: 1 }, TURNING_TIME / 2);
    new FlxTimer().start(1,function(timer){

        if(PlayState.questionCard.cardIndex!=cardIndex){

            PlayState.blackscreen.visible = true;
            PlayState.gameoverText.visible = true;
            FlxG.sound.play("assets/sounds/GameOver.wav");
            new FlxTimer().start(0.5, function (timer)
            {
                FlxG.switchState(new MenuState());
                //FlxG.resetState();
            });
            }
        else{
            PlayState.cardcount+=1;
            PlayState.questionCard.kill();
            this.kill();
            if(PlayState.cardcount>=12){
                PlayState.blackscreen.visible = true;

                PlayState.gamewinText.visible = true;
                FlxG.sound.play("assets/sounds/applause_y.wav");
                new FlxTimer().start(1, function (timer)
                {
                FlxG.switchState(new MenuState());
                //FlxG.resetState();
                });

            }
            PlayState.isquestionCardOpen=0;


        }
});

}
```

Figure 9: Event Handling for ans card

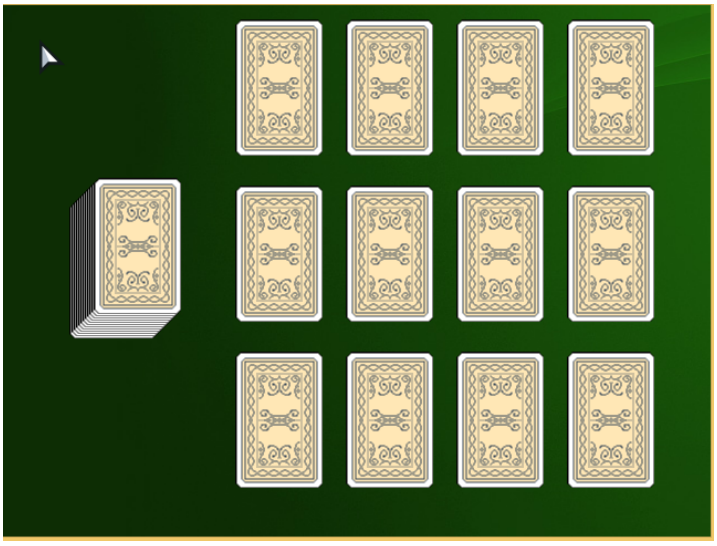
VI. Output

Menu screen

Game Screen



After 10 sec



After few Moves